Universidad Autónoma de Madrid

Escuela Politécnica Superior

Grado en Ingeniería Informática

TRABAJO DE FIN DE GRADO

PROGRAMACIÓN POR BLOQUES PARA UN ROBOT MÓVIL

Irene Colmenar Guindal
Tutor: Luis Fernando Lago Fernández

Junio 2016

PROGRAMACIÓN POR BLOQUES PARA UN ROBOT MÓVIL

Autor: Irene Colmenar Guindal
Tutor: Luis Fernando Lago Fernández

Departamento de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid

Junio 2016

Abstract

Abstract — Over the last few years, technology has changed our life without us even noticing it, and from the beginning we wanted to teach kids the right way to use it. This formation was initially mainly passive, allowing students to just experiment with tools, which did not fulfill their curiosity. However, in the last few years this method has been changing into a more active one, where kids are able to create apps and learn what is behind those apps that they use everyday.

To help with this new kind of teaching many proposals, like Scratch or Code, have been launched that allow us to teach coding to children in a more approachable way by using a programming language based on blocks. The main problem with these approaches is that they provide just a way of coding with a virtual solution. So, to make it even more attractive and interactive we have proposed the developement of a coding language based on blocks to program a robot built with an Arduino board.

The developement of this tool has been done by using the block's coding library "Blockly" released by Google. The customed blocks developed, have been designed for kids of young ages that are still not ready to read fluently. This blocks' design was done by using only pictures to explain their meaning. There are also blocks designed for older kids that allow them to do iterative codes and conditionals. The robot's movement has been acomplished by using a robot control tool developed in Python that gets the robot to the coordinates indicated.

Once the last operative prototype was developed, succesfull tests were run in a real environment with users aged between 6 and 8 years. These tests have demonstrated the app's usability for kids of younger ages, and the requirements established at the beginning of the project have been considered fulfilled.

Key words — Web App, Graphical Programming Blocks, Teaching-Learning Environment, Arduino

Resumen

Resumen — En los últimos años la tecnología ha ido cambiado nuestro día a día sin apenas darnos cuenta y, desde el principio, hemos querido impartir una formación a los más pequeños para su correcto uso. Esta formación empezó siendo principalmente pasiva, permitiendo a los alumnos sólo experimentar con herramientas ya desarrolladas que no parecían terminar de satisfacer su curiosidad. Sin embargo, en los últimos años, se ha ido cambiando esta formación por una mucho más activa en la que al alumno se le permite crear aplicaciones y aprender qué es lo que hay tras esas aplicaciones que usan en su día a día.

Para ayudar a este nuevo tipo de enseñanza, se han lanzado múltiples propuestas como Scratch o Code que nos permiten poner en contacto a los más pequeños con la programación de una forma mucho más cercana, mediante un lenguaje de programación por bloques. El principal problema de estas aplicaciones es que no dejan de ser una programación con un resultado visualizable de una forma virtual. Para hacer la programación aún más atractiva y tangible hemos planteado la realización de un software de programación por bloques para un robot imprimible controlado por una placa Arduino.

La realización de esta herramienta se ha realizado integrando el lenguaje de código libre de programación por bloques creado por Google, "Blockly". Los bloques desarrollados se han orientado para niños de corta edad que aún no saben leer teniendo en cuenta un diseño de los mismos con solo imágenes y algunos más avanzados que permitan realizar códigos iterativos y condicionales. Para el movimiento del robot se ha utilizado una herramienta de control desarrollada en Python que traslada al robot a las coordenadas que se le indiquen.

Una vez creado el último prototipo funcional del proyecto, se han realizado pruebas con éxito en un entorno real con usuarios de edades entre 6 y 8 años. Tras su realización, se ha podido comprobar la usabilidad de la aplicación para usuarios de edades tempranas y se han considerado cumplidos los objetivos propuestos durante la definición inicial del proyecto.

Palabras clave — Aplicación web, Lenguaje Programación por Bloques, Entorno Enseñanza-Aprendizaje, Arduino

Acrónimos

AJAX Asynchronous JavaScript And XML. 11

CSS Cascading Style Sheets. 12

DOM Document Object Model. 11, 12, 24, 30

HTML HyperText Markup Language. 11

S4A Scratch For Arduino. 6

SVG Scalable Vector Graphics. 12, 30

TIC Tecnologías de la Información y la Comunicación. 1

W3C World Wide Web Consortium. 11

Índice general

Índice de tablas

Índice de figuras

1 | Introducción

En este capítulo se detalla una breve descripción de los motivos que han llevado a la realización de este proyecto y sus objetivos principales. A continuación, se indica la estructura que sigue el presente documento.

1.1. Motivación

En los últimos años la tecnología ha ido cambiado nuestro día a día. Sin darnos cuenta se ha convertido en una parte vital de nuestro trabajo, de nuestra manera de relacionarnos e incluso de nuestra forma de pasar el tiempo [1].

La inclusión de la tecnología en ámbitos como la medicina o la ciencia han hecho posible la resolución de múltiples investigaciones [2]. Por todo esto, se considera vital la incorporación de las Tecnologías de la Información y la Comunicación (TIC) en las aulas de los colegios, haciendo necesario el desarrollo de herramientas para llevarlo a cabo.

Hasta ahora, en los colegios se estaba realizando esta incorporación de una forma pasiva en la que los alumnos eran meramente usuarios de distintas aplicaciones que hacían de complemento a los libros y cuadernos habituales. Sin embargo, en los últimos años se está insistiendo en la necesidad de añadir una forma de enseñanza activa en la que se fomente su curiosidad en la creación de tecnología [3].

La idea principal seguida para incluir el concepto de creación es la incorporación de la programación en las aulas.

A día de hoy, son pocos los que se paran a preguntar cómo está hecha una página web o una aplicación, y aún menos los que buscan la respuesta. Esta mitificación de la programación fue el principal motivo para el desarrollo de un proyecto que acercase a niños al entendimiento del código que hay por debajo de aplicaciones.

Partiendo de la idea de acercar la programación a los más pequeños, es difícil ignorar la fascinación que les crea la utilización de las TIC y el atractivo que estas suponen. En la actualidad existen multitud de herramientas que, utilizando lenguajes de programación por bloques, hacen la programación mucho más accesible a los más pequeños, pero no deja de ser una programación virtual.

Para hacer la programación aún más atractiva y visual están incluyéndose robots en las aulas. Un problema con el que se están encontrando, es la complejidad en su programación. Por este motivo, su incorporación sólo se está realizando a partir de edades más avanzadas.

Pensando en las múltiples posibilidades que puede aportar el control de un robot en su

inclusión en el curriculum escolar desde edades tempranas surge la idea del desarrollo de una herramienta que acerque su programación con un lenguaje cuya sintaxis esté diseñada usando bloques y además aprovechar su atractivo para proponer ejercicios que necesiten su uso para resolverlas, aportando así a profesores y padres nuevos recursos de enseñanza.

1.2. Objetivos

El objetivo principal de este proyecto es la realización de lenguaje de programación adaptado a niños que permita el control directo de un robot. Para ello, se desarrollará una herramienta web que incluya la plataforma de programación y actividades que aprovechen el atractivo del robot para el aprendizaje de otros contenidos. Este proyecto se considerará realizado cuando se cumplan las siguientes cuestiones ordenadas según su relevancia:

1. Desarrollo de un **lenguaje de programación** de robots Arduino orientado a niños.
2. **Entorno de programación** visual.
3. **Herramienta web** con distintos ejercicios cuyo método de resolución se realice mediante el lenguaje de programación desarrollado.
4. Ejercicios **clasificados por la edad** a la que están orientados.
5. **Instrucciones** de programación **disponibles** en el entorno de programación adecuados al nivel indicado en la clasificación de los ejercicios.
6. Existencia de una jerarquía de roles **profesor/alumno** en el sistema, en la que el profesor puede seguir el avance de los alumnos.
7. Inclusión de un sistema para los usuarios de **creación de nuevos ejercicios**.
8. Diseño de la herramienta **atractivo e intuitivo**.
9. Variedad de **ejercicios** que utilicen el control del robot para la enseñanza de programación y aprovechen el atractivo que supone un **robot móvil como sistema de apoyo** en otros ámbitos de la enseñanza.

1.3. Estructura del documento

- En el **capítulo 2** se exponen el estado del arte, en el que se analizan las herramientas y los métodos que se usan en la educación actualmente para la educación en las nuevas tecnologías.
- En el **capítulo 3** se exponen la definición del proyecto, la metodología seguida en su realización y las herramientas utilizadas para su desarrollo.
- En el **capítulo 4** se expone el análisis del proyecto realizado en el que se definen los requisitos funcionales y no funcionales que se han considerado para el posterior desarrollo.
- En el **capítulo 5** se exponen el diseño considerado para el desarrollo y las técnicas seguidas durante su elaboración.
- En el **capítulo 6** se exponen las pruebas realizadas y los resultados obtenidos
- En el **capítulo 7** se exponen las conclusiones obtenidas tras la realización de este proyecto.

2 | Estado del Arte

En este capítulo se realiza un estudio de algunas de las tecnologías actuales que se relacionan con este proyecto. En este estudio se concluyen las ventajas e inconvenientes de cada una para poder extraer ideas que puedan servir para la contextualización del proyecto.

2.1. Introducción

Desde que surgió el interés por enseñar a programar a niños, han ido apareciendo múltiples propuestas para introducir la programación a los más pequeños.

2.2. Herramientas existentes

A continuación se analizan algunas de las herramientas más populares de la actualidad.

2.2.1. Code

Code [4] es una herramienta web que nació como resultado de un estudio en el que se revelaba que sólo un 10 % de las escuelas de EEUU enseñaba programación en las aulas aún cuando ya era considerada un campo fundamental. Su objetivo principal fue la creación de una base de datos de todas las clases de programación de las escuelas de Estados Unidos.

Para desarrollar su iniciativa, Code utiliza la librería Blockly [5] para crear un lenguaje de programación visual basado en bloques con el objetivo de enseñar la lógica de la programación antes de la gramática que esta conlleva. Esta decisión se basa en la premisa de que, aunque cada lenguaje de programación es distinto, la mayoría siguen una misma lógica, la ejecución de instrucciones de forma secuencial, incluyendo la repetición de estas instrucciones o la ejecución condicional de una u otra.

Code además incluye una dinámica alumno/profesor muy interesante a la hora de la integración curricular de esta herramienta de tal forma que el profesor puede seguir el avance de cada alumno de forma personalizada.

Una ventaja de Code respecto de otras herramientas, es el número de ejercicios que almacena y la clasificación de éstos según la edad de los usuarios, siguiendo así un modelo

de realización incremental para que puedan seguir un curso como si de un temario se tratase.

Pero quizá, al abrir Code lo que más llame la atención es el diseño de su aplicación. Esta herramienta web atrae al usuario desde el momento en que la abre, teniendo en sus ejercicios a los personajes de juegos y películas favoritos de los niños, lo que hace que estos la vean, no como una actividad más con la que estudiar, si no como una web de juegos con los que irán aprendiendo la lógica de la programación sin darse cuenta.

2.2.2. Scratch

Scratch [6] es un lenguaje de programación visual libre creado para su dedicación a la enseñanza.

Parte de una premisa distinta a la de Code. Mientras que Code se centra en ejercicios que los usuarios tengan que resolver, Scratch le da más rienda suelta a la imaginación de los niños brindándoles las herramientas que puedan necesitar para que, mediante el tratamiento de imágenes y sonidos, puedan crear todo tipo de actividades.

El lenguaje de programación visual que sigue Scratch está basado en bloques con un modelo "drag & drop", es decir, de arrastrar y soltar.

Un inconveniente de Scratch es la dificultad de su uso para niños que aún tienen cierta dificultad para la lectura, además de la ausencia de un entorno de enseñanza con roles profesor/alumno que podría facilitar la inclusión de la herramienta en las aulas.

Existe una modificación de Scratch, Scratch For Arduino (S4A) [7] que permite programar las placas Arduino mediante el lenguaje de programación de bloques planteada por Scratch. El uso de este lenguaje está orientado a niños a partir de los 12 años y la exactitud del resultado de la programación depende, en gran medida, del montaje del robot.

2.2.3. Bitbloq

Bitbloq [8] es la nueva iniciativa de BQ [9] realizada para facilitar la programación de sus placas Arduino. Siguiendo la idea de programación visual, bitbloq trata los distintos accesorios que pueden agregarse a la placa Arduino como piezas en su entorno, indicando su inclusión en la placa de una forma muy visual.

La programación sigue la dinámica de Scratch, libre creación y programación del robot y el inconveniente de que sólo está orientado a niños que ya saben leer.

Una ventaja que tiene bitbloq en cuanto a la orientación de programación de Arduino, es la posibilidad de ver el código de bloques traducido a lenguaje Arduino, de tal forma que los usuarios pueden ir comparando el código programado por bloques con el código real de las placas.

Esta herramienta también permite al usuario guardar sus creaciones para poder seguir el programa en cualquier otro momento, y, basándose en el almacenamiento en la nube, hace que sea fácil e intuitivo para el usuario.

2.2.4. LEGO Mindstorms

LEGO Mindstorms [10] es una plataforma educativa de robótica lanzada por LEGO que combina la construcción de robots con la programación de los mismos.

La programación de los robots se realiza mediante un lenguaje de programación por bloques orientado a niños a partir de los 10 años, que permite el movimiento de motores, la obtención de datos de sensores e instrucciones más avanzadas como bucles, condicionales o la creación de funciones.

La mayor desventaja de estos productos es su alto precio además de que el lenguaje de programación que utilizan sólo es compatible con sus productos.

2.3. Conclusiones

En todas las herramientas estudiadas, el lenguaje de programación que utilizan se basa en bloques con un modelo de programación "drag & drop". Mientras que Code y Scratch utilizan esta forma de programación para la simulación de ejercicios, bitbloq y LEGO Mindstorms combinan la programación por bloques con el control de robots, lo que añade un atractivo más a la programación y además permite a los usuarios desarrollar su creatividad.

Otra idea destacable es la plataforma de educación planteada por Code en la que los niños tienen una serie de ejercicios propuestos para resolver que siguen un modelo de dificultad incremental, proponiéndoles un reto detrás de otro.

La librería Blockly que utiliza Code es una herramienta libre creada por Google que da todo tipo de funciones para crear con ella bloques personalizados.

Teniendo en cuenta la idea principal de este proyecto de la realización de un lenguaje de programación para el control de placas Arduino, las propuestas de bitbloq y LEGO Mindstorms son muy interesantes, pero estudiándolas hemos visto un problema a la hora de introducir este tipo de proyectos a niños pequeños debido a la complejidad del diseño de sus bloques.

Otro inconveniente encontrado con este tipo de propuestas es la inexactitud que pueden tener ambas herramientas a la hora de realizar movimientos siguiendo una programación muy básica que puedan realizar niños pequeños. Para resolver este inconveniente, la programación de esta herramienta se realiza siguiendo un modelo de ejercicios basados en una cuadrícula donde los movimientos del robot se harán de casilla en casilla. Además, para asegurar la exactitud del movimiento del robot se integra en el proyecto un sistema

de control visual basado en [11].

Se ha observado como factor común en todas las herramientas analizadas la importancia que tiene el diseño de la interfaz gráfica de una herramienta a la hora de captar la atención de los niños para animarles a aprender.

Para solucionar el alto coste de los robots que suponen las plataformas como Mindstorms, los robots imprimibles controlados por una placa Arduino son una buena alternativa.

Teniendo estas ideas en cuenta, la herramienta que se desarrolla en este proyecto reuniría todos los aspectos positivos destacados en el estudio anterior que permitan a niños iniciarse en la programación de robots:

- Programación por bloques de un robot construido con una placa Arduino.
- Diseño de los bloques de control del robot más sencillo que con bitbloq o LEGO Mindstorms.
- Posibilidad de ejecución simulada de los programas y ejecución real en el robot.
- Plataforma de enseñanza similar a la ideada por Code, con usuarios de tipo profesor y alumno, y control del progreso de los alumnos.

3 | Definición del proyecto

En este capítulo se explica una visión general del proyecto y la metodología seguida durante su desarrollo. A continuación, se incluyen las tecnologías y herramientas utilizadas.

3.1. Visión general

El sistema que se describe en este proyecto es el desarrollo de un lenguaje de programación por bloques para el control de un robot Arduino integrado dentro de una aplicación web diseñada para la enseñanza de niños. Para apoyar esta enseñanza, la herramienta sigue una dinámica profesor/alumno que permite al docente seguir el avance del alumno en la aplicación y diseñar ejercicios personalizados para su clase. La resolución de los ejercicios de programación se realiza usando un lenguaje de programación visual personalizado basado en una estructura de bloques.

Las actividades con las que cuenta la herramienta están clasificadas según la edad para la que estén dirigidas pudiendo seleccionarse ejercicios aptos para niños que aún no saben leer. Los bloques disponibles en cada actividad dependen del nivel al que pertenezcan, quedando establecido un modelo incremental de resolución de los ejercicios. La asignación de nivel en los ejercicios se realiza en su creación, permitiendo al usuario que diseñe el ejercicio designar la dificultad del mismo.

Debido a la edad de los usuarios a los que está destinada la aplicación, los ejercicios de programación propuestos están basados en una **cuadrícula** en la que cada celda es una **casilla** a la que se puede mover el robot, siendo los movimientos de casilla en casilla en las direcciones norte, sur, este y oeste.

La resolución de los ejercicios puede realizarse de forma simulada en la aplicación o realizando la ejecución en el robot. La simulación visualiza una animación del robot moviéndose por la cuadrícula según los movimientos indicados con la programación por bloques.

El sistema también incorpora diferentes actividades que incluyen el uso del robot. Estas actividades siguen un modelo de pregunta/respuesta en el que el sistema propone preguntas y el usuario va respondiéndolas señalando la respuesta correcta en la aplicación de una forma interactiva.

Aunque el tipo de actividades desarrollables es muy amplio, en este proyecto se desarrollan dos de tipo geográfico, en la que el usuario selecciona la respuesta en un mapa. Al indicar la respuesta, el robot se desplaza a la posición indicada en el mapa,

indicando en la interfaz web si la respuesta ha sido correcta.

El movimiento del robot se realiza utilizando el software desarrollado por Iván Márquez Pardo para su Trabajo de Fin de Grado en la Universidad Autónoma de Madrid [11]. Este software, se encarga de trasladar al robot a una posición corrigiendo los movimientos del robot mediante la imagen obtenida por una cámara. El desarrollo de ambos proyectos se ha realizado paralelamente teniendo en cuenta la posterior integración de ambos softwares.

3.2. Metodología

La planificación del proyecto sigue un ciclo de vida incremental iterativo compuesto por incrementos divididos en 6 etapas de desarrollo:

1. Análisis
2. Diseño
3. Codificación
4. Pruebas Unitarias
5. Pruebas de Integración
6. Instauración y aprobación del sistema

Este modelo de ciclo de vida permite tener una versión estable del programa desde el desarrollo del primer incremento gracias al modelo de cascada que se realiza en cada iteración que permite obtener versiones estables en la finalización de cada incremento. Además, utilizando el modelo incremental, los requisitos pueden categorizarse según su prioridad pudiendo añadirlos al proyecto tras la realización de cada incremento.

Según esta clasifación, los incrementos serán los siguientes:

- Iteración 1: Subsistema de ejercicios de programación.
- Iteración 2: Subsistema de traducción de ejercicios.
- Iteración 3: Subsistema de almacén de ejercicios.
- Iteración 4: Subsistema de integración del software de control del robot con la aplicación web.
- Iteración 5: Susbsistema de gestión de usuarios.
- Iteración 6: Subsistema de creación de tableros.
- Iteración 7: Subsistema de actividades adicionales.

3.3. Tecnologías y herramientas utilizadas

A continuación se listan las tecnologías utilizadas en el desarrollo de la herramienta web:

- HTML5 [12] en la codificación de la interfaz de la herramienta web. Es la quinta revisión del lenguaje estándar HyperText Markup Language (HTML) realizado por la World Wide Web Consortium (W3C) para la elaboración de páginas web.
- CSS3 [13] en la codificación del estilo de la página web. Es la última versión del lenguaje utilizado para definir y crear la presentación de un documento escrito en HTML.
- JavaScript [14] para la animación gráfica y la parte lógica de la aplicación. Es un lenguaje de programación interpretado basado en el estándar ECMAScript. Su aplicación en las web está orientado tanto al lado del cliente para la dinamicidad de la web como en el lado del servidor para relizar la parte del back-end.
- JQuery [15] para la animación gráfica. Es una biblioteca de JavaScript que permite simplificar la interacción con los documentos HTML, manipular el árbol Document Object Model (DOM), desarrollar animaciones y realizar cambios sobre las páginas sin necesidad de recargarlas mediante la técnica Asynchronous JavaScript And XML (AJAX).
- MySQL [16] en la base de datos para el almacenamiento de usuarios, tableros y progresos. MySQL es un sistema de gestión de base de datos relacional.
- PHP [17] para el tratamiento de datos entre las distintas vistas de la página web y ejecución del código Python asociado a la comunicación con el robot. Este lenguaje se usa principalmente para el desarrollo web de contenido dinámico.
- Python [18] para la transmisión de datos entre el sistema web y el software de control del robot. Python es un lenguaje de programación interpretado de orientación a objetos cuya sintaxis favorece un código legible.

Las herramientas que se han utilizado durante el desarrollo han sido las siguientes:

- Sublime Text [19] como entorno de programación.
- Sequel Pro [19] para la gestión de la base de datos.
- Adobe Illustrator [20] para el tratamiento de imágenes vectoriales.
- Bitbucket [21], basado en la tecnología Git para el control de versiones.

Los recursos gráficos se han obtenido de la la web Freepik [22], un buscador de recursos gráficos gratuitos para diseñadores.

Los iconos utilizados en la aplicación se han obtenido de Font Awesome [23], una fuente de iconos vectoriales tratables mediante Cascading Style Sheets (CSS).

Además, se han usado cuatro herramientas externas que detallamos a continuación:

3.3.1. Blockly

Blockly [5] es una librería codificada en JavaScript y es la principal idea de la que partió este proyecto, un lenguaje visual por bloques que ayudase a cualquier usuario de la aplicación web a aprender la lógica de la programación de una forma mucho más accesible y atractiva para un usuario que se encuentra en su primera toma de contacto con la programación.

Esta biblioteca nos permite la realización de bloques personalizados y la definición del código que generarán al ser ejecutados, brindándonos así de múltiples posibilidades a la hora de utilizarla.

3.3.2. RaphaëlJs

RaphaëlJS [24] es una librería de JavaScript para simplificar el trabajo con imágenes vectoriales en la web. La funcionalidad principal que brinda esta librería es la posibilidad de tratar todos los objetivos gráficos como objetos DOM que pueden asociarse a JavaScript handlers.

Su funcionamiento consiste en el manejo de canvas basado en el estándar SVG [25]. Este estándar describe una secuencia de nodos que son utilizados por el framework para facilitar el trabajo con vectores gráficos en la web.

3.3.3. C3.js

C3 [26] es una librería basada en D3 [27] que permite la integración de gráficas en aplicaciones web.

Esta librería asigna clases a cada elemento al generarlas para poder definir un estilo único a cada clase. Además, permite seleccionar los datos que se quieren visualizar en cada momento desde la interfaz gráfica.

3.3.4. Software de control del robot

Para enviar las instrucciones al robot se utiliza el sistema desarrollado por Iván Márquez Pardo en su TFG “Control visual de un robot móvil mediante una cámara

cenital" [11]. Este sistema realiza el control de un robot móvil mediante el seguimiento del movimiento del mismo en una imagen capturada por una cámara.

Debido a que la resolución de los ejercicios se realiza en un tablero cuadriculado, es necesario que los movimientos que el robot haga sean precisos y el usuario no tenga que preocuparse de desviaciones que pudiesen producirse por el entorno o la construcción del robot.

Con la utilización de este código se ha podido asegurar la correcta ejecución de las instrucciones de movimiento del robot que se indican en la programación por bloques, asegurando que el robot termina tras la ejecución de cada instrucción en la casilla correcta.

4 | Análisis

En este capítulo se definen los requisitos del proyecto. Para la especificación de requisitos considerados para el proyecto se han definido distintos subsistemas que los engloban:

- **Gestión de usuarios**: Subsistema encargado de la gestión del registro e inicio de sesión de los usuarios además de los datos asociados a cada usuario.
- **Creación de Tableros**: Subsistema encargado de la creación de ejercicios.
- **Almacén de ejercicios**: Subsistema encargado de la gestión y clasificación de ejercicios.
- **Ejercicios de programación**: Subsistema encargado de la resolución y validación de ejercicios.
- **Interacción con el robot**: Subsistema encargado de la traducción de ejercicios del idioma de bloques a coordenadas que envía al robot y de la comunicación con el mismo.
- **Actividades adicionales**: Subsistema encargado de las actividades adicionales que siguen un modelo pregunta/respuesta.
- **Interfaz y usabilidad**
- **Rendimiento**
- **Requisitos técnicos**
- **Portabilidad**

4.1. Requisitos funcionales

Gestión de usuarios

RF 1. El sistema permite la creación de cuentas de usuarios.

RF 2. Las cuentas de usuario pueden ser de dos tipos: profesor o alumno.

- **RF 2.1.** El registro de usuarios requiere del nombre, apellidos, cuenta de correo electrónico, contraseña del usuario y rol en la aplicación.
- **RF 2.2.** Si el usuario es profesor, podrá añadir y eliminar usuarios asociados a su cuenta.
- **RF 2.3.** La cuenta de correo es única para cada usuario, no pudiendo esta ser usada para más de un usuario.
- **RF 2.4.** La identificación del usuario en el sistema se realiza con su dirección de correo electrónico y la contraseña indicada en el registro.

RF 3. Desde la cuenta de cada usuario se puede visualizar los tableros que este ha creado.

Creación de Tableros

RF 4. El sistema permite crear mapas de tipo recorrido o libre, diseñados en un tablero cuadriculado donde cada celda es una casilla por la que se puede mover el robot.

- **RF 4.1.** Los mapas de tipo recorrido sólo son resueltos siguiendo un itinerario, que describe el recorrido que tiene que seguir el robot desde una casilla inicial hasta la meta.
- **RF 4.2.** Los mapas libres contienen las casillas por las que puede y no puede pasar el robot, además de la casilla inicial y la casilla final.

RF 5. En la creación de los mapas, la primera casilla que se selecciona es la casilla inicio, en la que se tiene que empezar la resolución de la actividad y, la última, la meta.

RF 6. En la creación de mapas, la modificación de los itinerarios se realiza seleccionando la última casilla marcada, así, se van borrando desde el final al principio, no siendo posible la modificación del recorrido por en medio.

RF 7. En los mapas libres, la reselección de una casilla la elimina del mapa.

Almacén de ejercicios

RF 8. En la página principal se visualizan los diferentes tipos de ejercicios que pueden realizarse.

RF 9. El usuario puede ver los ejercicios disponibles dentro de cada tipo seleccionándolo.

RF 10. Dentro de cada tipo los ejercicios se clasifican por nivel.

Ejercicios de programación

RF 11. La resolución de ejercicios se realiza mediante un lenguaje de programación personalizado basado en programación por bloques.

RF 12. Las instrucciones disponibles son dirección de movimiento del robot, bucles y condicionales.

RF 13. Los bloques disponibles en cada ejercicio dependen del tipo y nivel del ejercicio a resolver.

- **RF 13.1.** Los ejercicios de nivel 1 tienen disponibles bloques de dirección.
- **RF 13.2.** Los ejercicios de nivel 2 tienen bloques de dirección y bucles.
- **RF 13.3.** Los ejercicios de nivel 3 tienen bloques de dirección, bucles y condicionales.

RF 14. El sistema tiene distintos tipos de ejecución de la solución de ejercicios.

- **RF 14.1.** Ejecución mediante simulación del código en la web.
- **RF 14.2.** Ejecución mediante el envío de movimientos paso a paso al robot.
- **RF 14.3.** Ejecución mediante el envío de la serie completa de movimientos programados al robot.

Interacción con el robot

RF 15. El sistema traduce las instrucciones del idioma blockly a coordenadas reales.

RF 16. El sistema se comunica con el software de control del robot y le envía las coordenadas generadas.

Actividades adicionales

RF 17. El sistema tiene actividades educativas que interactúan con el robot.

RF 18. La resolución de estas actividades será mediante la selección de la respuesta en la web sobre un mapa interactivo.

4.2. Requisitos no funcionales

Interfaz y usabilidad

RNF 1. La aplicación tendrá un diseño web adaptativo, adaptable a las distintas pantallas en las que la web se visualice.

RNF 2. El diseño de la web es atractivo y fácil de usar para los más pequeños, usando un diseño con carácter infantil e intuitivo.

RNF 3. Los textos de la web están disponibles en castellano.

Rendimiento

RNF 4. El tiempo de respuesta de la aplicación no debe llegar a ser superior a 5 segundos.

Requisitos técnicos

RNF 5. La web es visualizable en los navegadores Google Chrome 50.0, Safari 9.1.

Portabilidad

RNF 6. Los datos de los ejercicios y de los usuarios son almacenados en una base de datos MySQL.

5 | Diseño y Desarrollo

En este capítulo se describe el diseño y desarrollo del proyecto basado en el análisis realizado en el capítulo anterior.

La estructura de la aplicación ha sido diseñada en los 10 módulos representados en la Figura 5.1.

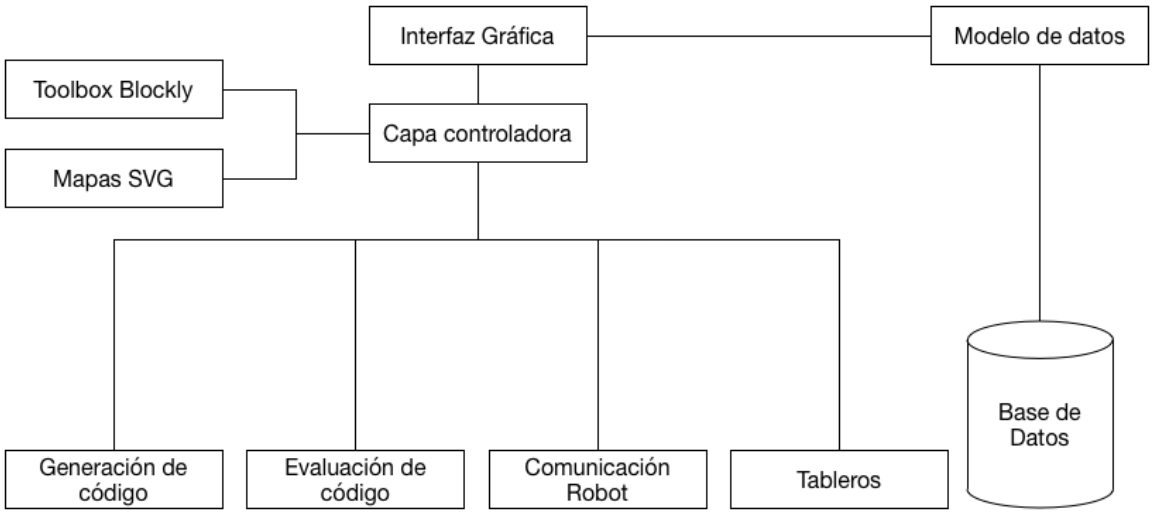


Figura 5.1: Esquema de la estructura de la herramienta

A continuación se describe de una forma detallada cada módulo diseñado.

5.1. Interfaz Gráfica

Es el módulo encargado de la creación de la interfaz de usuario. Para el desarrollo de la interfaz se utiliza un entorno web.

5.1.1. Usabilidad

En el desarrollo de la interfaz gráfica se ha tenido muy en cuenta el tipo de usuarios al que está orientada la aplicación, para ello se han seguido distintos factores de usabilidad extraídos de un artículo de "Diseño web orientado a niños" que puede encontrarse en [28]. Los factores tenidos en cuenta son los siguientes:

1. Se ha comprobado que los **sonidos y las animaciones** captan la atención de los niños y les motivan en la interacción con los sitios web y a alcanzar los objetivo planteados en los ejercicios que se proponen.

2. Debido a que los niños tienden a recorrer con el puntero del ratón las webs, se tiende a generar **elementos con efectos** que se produzcan cuando se pasa por encima de ellos.

3. Las **metáforas de navegación**, conceptos y modelos del mundo real, son necesarias en este tipo de diseño. El uso de iconos como flechas de reset, iconos de guardado o de reproducción ayudan a los usuarios al rápido entendimiento y a la asociación de la acción que realizan los botones que puedan aparecer durante la navegación. La principal razón de la usabilidad de iconos en este tipo de diseños es la menor necesidad de capacidad de lectura para su entendimiento.

4. Se sugieren **páginas cortas** debido a la no tendencia de los niños a utilizar la barra de scroll.

5. Debido a que los menores tienden más a leer las instrucciones de uso que los adultos, las **secciones de ayuda** son recomendadas teniendo siempre en cuenta la sintetización de los textos que las componen.

5.1.2. Navegabilidad

La navegación en la web se ha estructurado según el esquema representado en la Figura 5.2.

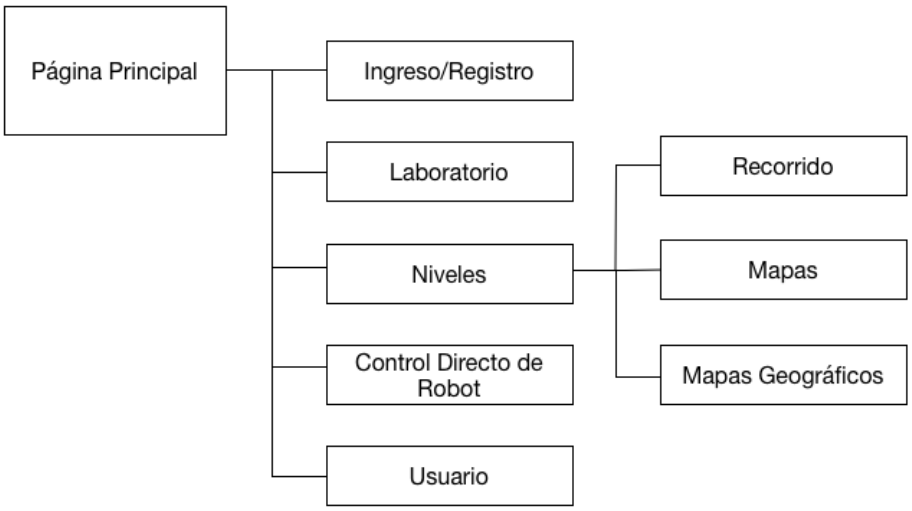


Figura 5.2: Esquema de la navegación por la página web

Para facilitar la navegación por la web, en las vistas se incluye un menú principal cuyo contenido depende de si se ha realizado previamente el ingreso en la aplicación. El menú tras el inicio de sesión es como el indicado en la figura 5.3.

Figura 5.3: Maqueta del menú principal

Antes del inicio de sesión, el menú sólo muestra las opciones de página de inicio e inicio de sesión.

Basándonos en el objetivo principal de la usabilidad que se ha definido en el apartado de análisis del proyecto, el diseño de las maquetas ha sido el indicado en los siguientes apartados. Todas las imágenes de las maquetas se encuentran en el apéndice B.

Página principal

Es la vista principal de la aplicación, en ella se muestran las opciones comunes del menú superior y un listado con las diferentes categorías de las actividades que existen: recorridos, control del robot, mapas y mapas geográficos.

En la figura B.1 se muestra una maqueta del diseño propuesto.

Registro/Ingreso

Es la página para realizar el registro e ingreso del usuario en el sistema. Una vez el usuario ingresa puede tener acceso a todas las funcionalidades. Aunque el uso de la página está orientado a niños pequeños, se considera que un adulto les guiará durante el uso de la aplicación y les realizará el registro en el sistema.

En la figura B.2 se muestra una maqueta del diseño propuesto.

Laboratorio

Es la vista de la página de creación de nuevos tableros. En esta página se pueden crear tableros personalizados de tipo recorrido y de tipo mapa.

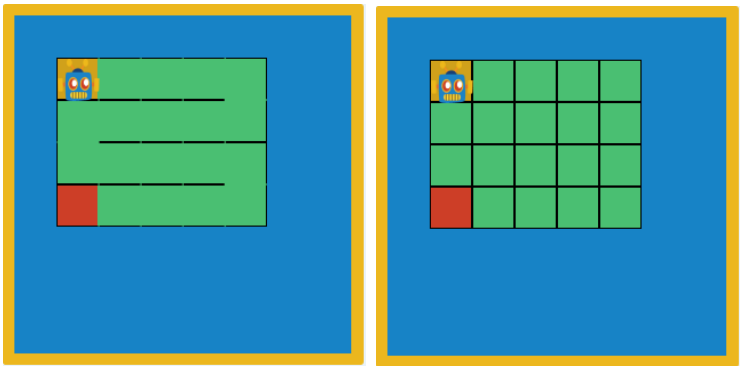
Los tableros de tipo **recorrido** tienen un itinerario de resolución predefenido, indicado mediante barreras para que pueda verse el camino que se debe seguir en su resolución. Los tableros de tipo **mapa** tienen como única regla la llegada a la casilla final, yendo siempre por casillas que sean transitables.

En la figura 5.4 se muestra la visualización de ambos tipos de tableros.

La vista de laboratorio está dividida en dos secciones principales: La primera contiene la cuadrícula de diseño de los mapas. La segunda sección, contiene las reglas de creación del tablero y un formulario de datos para crear el mapa y guardarlo en el tablero, en el que se indica principalmente si el tablero a crear es de tipo recorrido o mapa, ya que las reglas

a aplicar en la creación son diferentes. Además, se indica el nivel al que pertenece el tablero propuesto y el número de bloques de programación recomendados para su resolución.

En la figura B.3 se muestra una maqueta del diseño propuesto.



(a) Recorrido (b) Mapa

Figura 5.4: Tipos de mapas

Usuario

Esta vista depende del tipo de usuario que haya ingresado en el sistema, los elementos comunes a mostrar son, los datos concretos del usuario y un listado de los tableros que él ha creado.

Si el usuario es un alumno, se muestra un apartado de progreso en el que puede ver su evolución. Y si el usuario es un profesor, se muestra el progreso de los alumnos que tiene asociados.

En la figura B.4 se muestra una maqueta del diseño propuesto.

Niveles

Esta vista muestra un listado de las actividades disponibles dentro del tipo que se seleccionó al abrir este panel. Siempre que existan, se mostrará un menú en el que se puede seleccionar los niveles o tipo de los ejercicios que se mostrarán. En el caso de los recorridos se muestran los niveles disponibles, y en el caso de la sección de Geografía los distintos mapas a resolver.

En la figura B.5 se muestra una maqueta del diseño propuesto.

Recorrido y Mapas

Esta vista tiene dos secciones principales:

La primera es un panel dividido en dos subsecciones: los bloques disponibles de programación y el panel de programación que seguirá un modelo "drag & drop". Los bloques de programación del panel izquierdo serán arrastrados al panel derecho para la resolución de los ejercicios.

En el panel de programación, los bloques a ejecutar se colocarán en modo cascada, encajándolos consecutivamente en vertical y dejando el resto del tablero limpio.

En la segunda sección se visualiza el nivel a resolver que se seleccionó en la ventana anterior, y los botones con las acciones disponibles:

- Simulación de código en la web.
- Ejecución del código en el robot paso a paso.
- Ejecución del código completo en el robot.
- Reinicio de la ejecución paso a paso.

En la figura B.6 se muestra una maqueta del diseño propuesto.

Control directo del Robot

En esta vista, se realiza una programación directa del robot sin tener un ejercicio propuesto. De esta forma, se pueden realizar diseños en el tablero real poniendo vallas o marcas en él para que el usuario lo resuelva desde la aplicación. La vista del robot se dividirá también en dos secciones: el panel de programación y los bloques de ejecución.

En la figura B.7 se muestra una maqueta del diseño propuesto.

Otras actividades

Las actividades adicionales propuestas en este proyecto para la integración del robot en otros ámbitos, han sido de tipo geográfico. Esta vista se compone de 3 secciones: un mapa interactivo donde se seleccionarán las respuestas, la pregunta propuesta para su resolución, y por último, un contador con los aciertos y los fallos cometidos durante la resolución del ejercicio.

En la figura B.8 se muestra una maqueta del diseño propuesto.

5.1.3. Interactividad de la web

Para añadir interactividad a las vistas web pudiendo mostrar y ocultar partes de la misma, se ha utilizado JQuery [15], un conjunto de librerías de JavaScript que nos

permiten la modificación del DOM en tiempo real. Gracias a estas peticiones, la página puede actualizar el contenido sin necesidad de salir de ella, pudiendo tener continuamente actualizadas secciones como los listados de alumnos, las preguntas en los ejercicios pregunta/respuesta o los aciertos y errores en estos mismos ejercicios.

Además, se ha desarrollado una animación de la ejecución de los bloques de código en el tablero, mostrándose una animación del avance del robot por el mismo para mostrar el recorrido desarrollado, pudiendo así detectar posibles errores cometidos durante la resolución.

5.2. Modelo de datos

Este módulo se ha desarrollado en PHP [17] y ha cubierto principalmente las siguientes funcionalidades:

- Tratamiento de datos

  Mediante variables de sesión, se ha mantenido en la parte PHP todos los datos necesarios de la web que no son necesarios para la lógica de la aplicación, principalmente los datos del usuario que está en uso de la aplicación.

- Interacción con la base de datos implementada en MySQL [16].

5.3. Base de datos

La base de datos desarrollada se ha diseñado siguiendo un modelo relacional compuesto por 4 tablas: Users, Boards, Progress y MapProgress.

La tabla **USERS** almacena los datos de los usuarios registrados en el sistema, su nombre, apellido, correo electrónico, contraseña codificada en el estándar md5, un identificador que indica si es profesor y un campo más para los alumnos para indicar el id del profesor al que puedan estar asociados.

La tabla **BOARDS** almacena los tableros con su información de creación, el tipo, el nivel al que pertenecen y el número de bloques recomendados para su resolución.

La tabla **PROGRESS** trata el progreso de cada usuario registrando las actividades que ha resuelto.

La tabla **MAPPROGRESS** trata el progreso de cada usuario en la resolución de mapas geográficos, esta tabla almacena el número de errores cometidos en cada resolución.

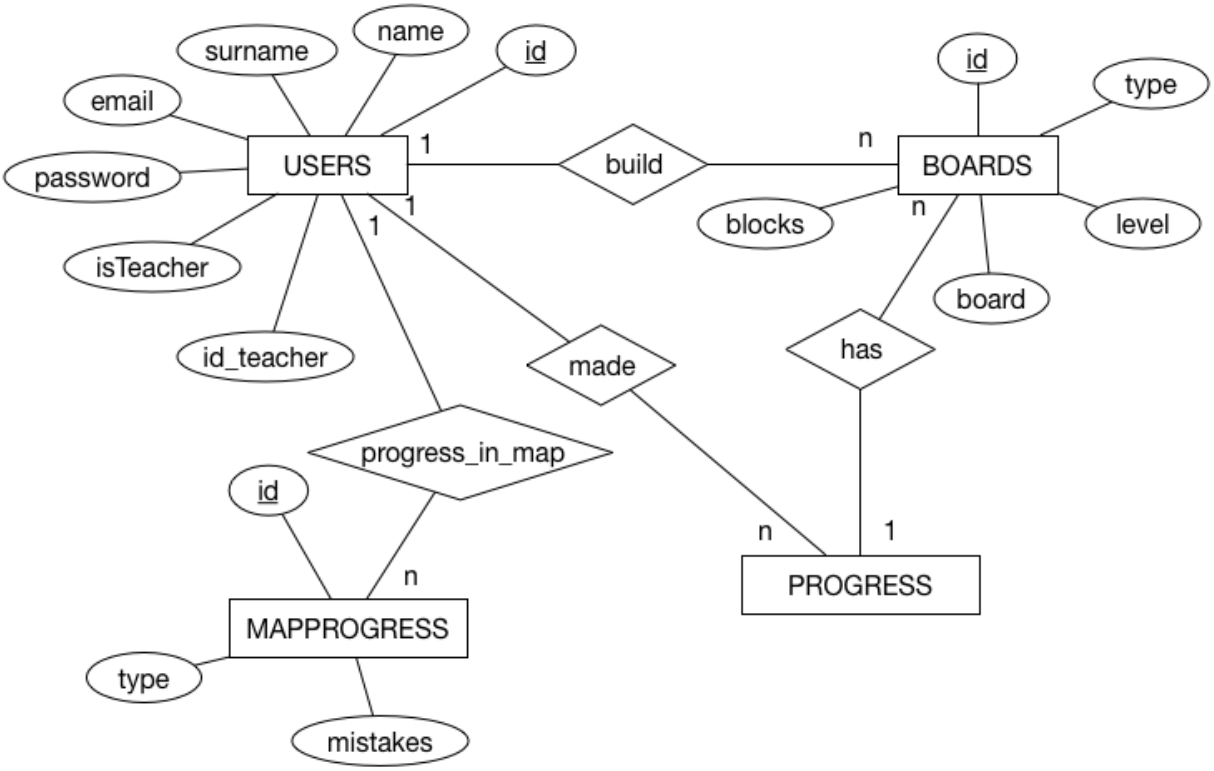


Figura 5.5: Diagrama Entidad Relación de la base de datos

5.4. Capa controladora

Es la capa que centraliza la lógica de la aplicación y realiza la interaccion de la interfaz gráfica con los distintos módulos de control.

Su funcionalidad principal ha sido la de resolución de ejercicios.

5.4.1. Resolución de ejercicios de tipo recorrido, mapa y envío directo al robot

Cada ejercicio muestra varias acciones de resolución, como se puede ver en la figura 5.6 tenemos 4 botones de ejecución:

Figura 5.6: Botones de ejecución: Play, Step, Download, Reset

- Play: Simulación del código en la web.

  - Generación de código: Traducción de los bloques de programación a movimientos.
  - Simulación del código en la animación.
  - Evaluación del código: Correcto o incorrecto.

- Step: Este modo envía paso a paso los movimientos generados a través del código indicado en la sección de programación.
  - Generación de código: Traducción de los bloques de programación a movimientos.
  - Envío de la posición de la casilla correspondiente en cada paso.

- Download: Descarga al robot la trayectoria completa generada a través de los bloques de programación indicados.
  - Generación de código: Traducción de los bloques de programación a movimientos.
  - Envío del código completo al robot.

- Reset: Reinicia la ejecución "paso a paso" y en el caso de ser un ejercicio simulado lleva la animación a la casilla inicial'.

En los modos de descarga de código en robot en los ejercicios de envío directo de código al robot se considerará la casilla inicial la esquina superior izquierda.

5.5. Generación de código

Este módulo genera el código indicado mediante la interfaz gráfica con los bloques Blockly. En la sección 5.9 se describe el funcionamiento de cada uno.

Cada bloque de dirección, tiene una sintaxis que se traduce a movimientos de una posición en las direcciones norte, sur, este y oeste.

El código de bucle generará el código resultante tras interpretar las iteraciones definidas.

Para la generación de código de tipo condicional, se interpreta el tablero que se esté ejecutando y, teniendo en cuenta los tipos de casillas de un tablero (transitable o no transitable), se genera el código con los movimientos resultantes.

5.6. Evaluación de código

Este módulo realiza un test sobre el código para comprobar si el ejercicio se ha solucionado correctamente.

Para realizar este test, se compara el código realizado en la programación por bloques con el itinerario del tablero de recorrido que se esté resolviendo.

Si el tablero es de tipo mapa se comprueba que todas las instrucciones son válidas y que la casilla final tras simular el código es la meta.

Al finalizar la evaluación se le indica al usuario si ha resuelto el ejercicio correctamente.

5.7. Comunicación Robot

La comunicación con el software del control del robot se realiza mediante la ejecución de scripts Python [18] llamados desde PHP.

Estos scripts son ejecutados al iniciar la ejecución en robot en la vista de nivel de un programa diseñado, y traducen las instrucciones generadas tras la interpretación de los bloques del panel de programación en coordenadas reales a las que deberá moverse el robot.

Las coordenadas se generan simulando una cuadrícula virtual sobre la imagen captada por el software de control del robot. Teniendo en cuenta el tamaño en pixeles de una casilla se transforman las posiciones de la cuadrícula en posiciones reales de la imagen.

Para la transmisión de las coordenadas de la aplicación web al sistema de control de robot se han utilizado tuberías definidas en la biblioteca "os" de Python, utilizando una tubería en la transmisión de información del sistema web al sistema de control y otra en la dirección opuesta.

En el siguiente código puede verse el procedimiento seguido para la creación y uso de las tuberías.

Código 5.1: Creación de tuberías para escritura

```
os.mkfifo(pipe_name, 0777) #Creacion de tuberia
pipeout = os.open(pipe_name, os.O_WRONLY) #Apertura de tuberia
    ↪ para escritura
os.write(pipeout, str1) #Escritura en tuberia
os.close(pipeout) #Cierre de tuberia
```

Código 5.2: Creación de tuberías para lectura

```
os.mkfifo(pipe_name) #Creacion de tuberia
pipein = open(pipe_name, 'r') #Apertura de tuberia para lectura
line = pipein.readline() #Lectura de tuberia
os.close(pipein) #Cierre de tuberia
```

5.8. Tableros

La programación por bloques se ha realizado siguiendo un diseño de movimientos dentro de una cuadrícula en la que cada celda es una casilla por la que el robot se irá moviendo.

Para el tratamiento de los tableros dentro del programa se ha realizado un diseño de programación orientado a objetos. El diseño seguido se muestra en la Figura 5.7.

Las clases de las que se compone son las siguientes:

- La clase **Level** es la contenedora de todos los tableros de un nivel.
- La clase **Board** representa los tableros y sus datos correspondientes, el nombre del tablero, el tipo, el nivel y el itinerario asociado.
- La clase **Itinerary** se forma por un conjunto de casillas.
- La clase **Cell** representa una casilla dentro del tablero mediante coordenadas x e y.

5.9. Toolbox Blockly

Para el diseño del lenguaje de bloques que se utiliza para la resolución de actividades y control del robot Arduino, usamos la librería Blockly. Con esta librería hemos diseñado 6 bloques personalizados y se ha desarrollado una generación de código para su posterior interpretación.

- Cuatro bloques de tipo **dirección** siguiendo los puntos cardinales, que indican el movimiento de una casilla en una dirección determinada.
- El bloque **bucle** itera el código que se indica dentro de él tantas veces como se seleccione en el argumento númerico que tiene.
- El bloque **condición** condiciona la ejecución de los bloques que se le indican como argumento si se cumplen las dos condiciones modificables que tiene, que son el tipo y la dirección de casilla consecutiva a comparar
    - Tipo: Las casillas pueden ser de dos tipos, transitable o no transitable (tierra o agua).
    - Dirección: Norte, sur, este y oeste.

En la figura 5.8 se muestra el diseño de los bloques.

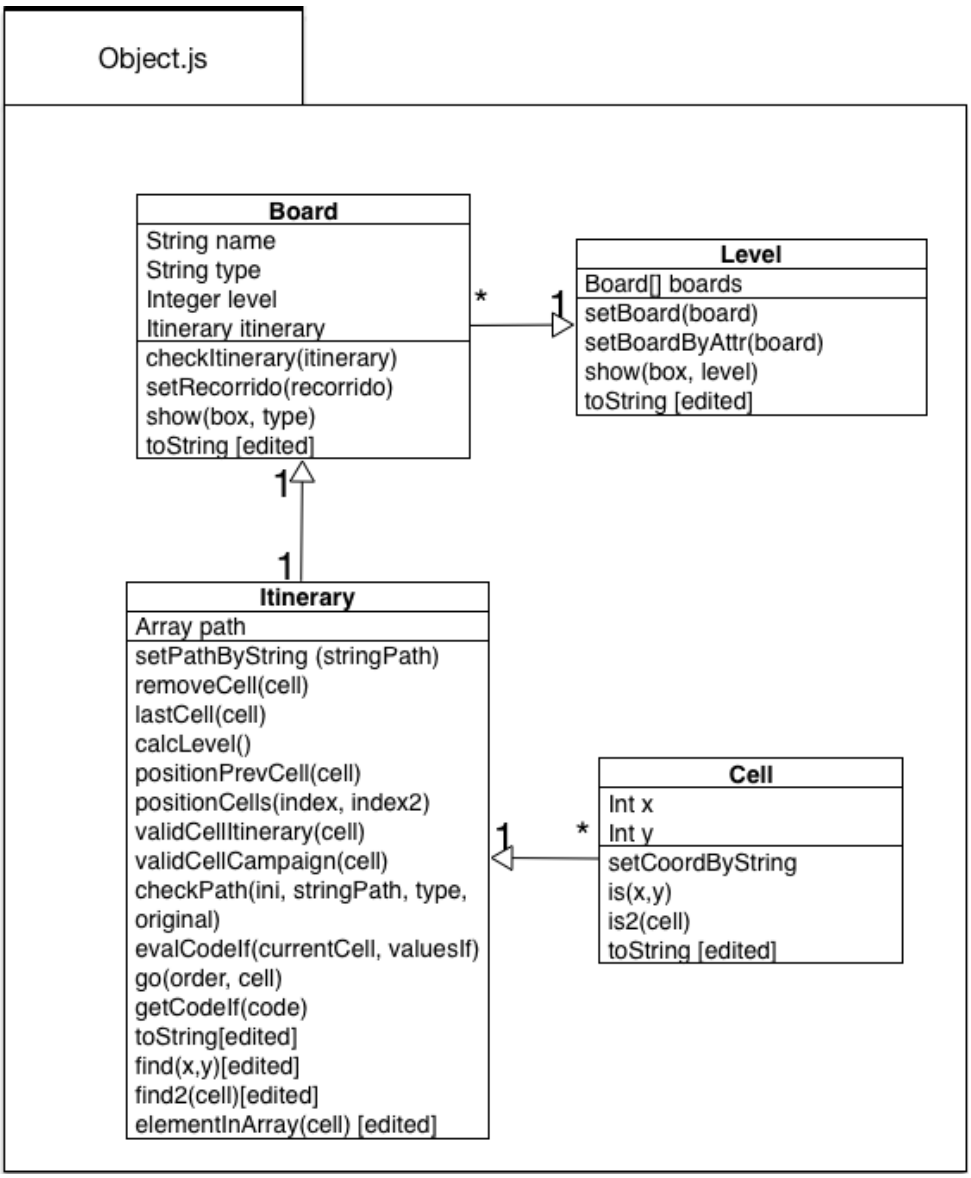


Figura 5.7: Diagrama de Clases de los tableros

Figura 5.8: Diseño de los bloques: Norte, Sur, Este, Oeste, Bucle, Condicional

5.10. Mapas SVG

Mediante la librería RaphäelJS, los mapas en formato SVG son tratados y traducidos a objetos DOM, esto permite el tratamiento de cada región del mapa como este tipo de objetos, que responden a eventos de clickado. Estos mapas son visualizados en la interfaz gráfica en la parte de actividades adicionales.

El desarrollo de este tipo de ejercicios se han hecho en su totalidad en JavaScript sin utilizar la base de datos debido a que las preguntas eran de geografía y las respuestas están prefijadas. Las preguntas acerca del mapa se lanzarán de forma aleatoria cada vez que se abra el nivel.

Las respuestas estarán asociadas a los objetos DOM, y al seleccionar una respuesta esta será validada. Cuando se finaliza un ejercicio, la aplicación se encargará de almacenar el progreso en la base de datos.

6 | Pruebas y resultados

En este capítulo se especificarán las pruebas que se han ido realizando a lo largo del desarrollo del proyecto y los resultados finales del proyecto.

En el apéndice C se pueden encontrar las imágenes de las versiones finales de la interfaz que se han utilizado durante las pruebas con usuarios.

Las pruebas realizadas han sido de dos tipos, en primer lugar pruebas para comprobar el correcto funcionamiento de la aplicación y en segundo lugar pruebas con usuarios.

6.1. Pruebas de funcionamiento de la aplicación

Nº	Descripción	Resultado esperado	Test
1	**Registro** en el sistema como **alumno** introduciendo correctamente todos los datos	Registro superado, base de datos actualizada e ingreso en el sistema realizado	Superado
2	**Inicio de sesión** en el sistema	Redirección a la página inicial con la sesión iniciada	Superado
3	**Visualización de ejercicios** sin haber sido asignado por ningún profesor ni haber creado ningún ejercicio	No aparece ningún ejercicio	Superado
4	**Cierre de sesión**	Redirección a la página de inicio con ninguna sesión iniciada	Superado
5	**Registro** en el sistema como **profesor** y asignación de alumno creado anteriormente	Alumno añadido correctamente	Superado
6	**Creación de un ejercicio** tipo recorrido de cada nivel, 1, 2 y 3	Ejercicios creados correctamente y almacenados en la base de datos	Superado
7	**Creación de un ejercicio** tipo mapa	Ejercicio creado correctamente y almacenado en la base de datos	Superado
8	Acceso con usuario alumno y **visualización de ejercicios** tipo recorrido	Existe un ejercicio de cada nivel	Superado
9	**Resolución** de un ejercicio tipo recorrido de nivel 1 con los **movimientos exactamente necesarios** para su resolución en modo **simulación**	El sistema valida el ejercicio y lo da por correcto	Superado

Nº	Descripción	Resultado esperado	Test
10	**Resolución** de un ejercicio tipo recorrido de nivel 1 con **más movimientos de los necesarios** para su resolución en modo **simulación**	El sistema valida el ejercicio y lo da por correcto	Superado
11	**Resolución** de un ejercicio tipo recorrido de nivel 1 con **movimientos erróneos** en modo **simulación**	El sistema valida el ejercicio y lo da por incorrecto	Superado
12	**Resolución** de un ejercicio tipo recorrido de nivel 2 con bloques de tipo bucle en modo **simulación** de forma correcta	El sistema valida el ejercicio y lo da por correcto	Superado
13	**Resolución** de un ejercicio tipo recorrido de nivel 3 con bloques de todos los tipos en modo **simulación** de forma correcta	El sistema valida el ejercicio y lo da por correcto	Superado
14	**Resolución** de un ejercicio tipo recorrido de nivel 3 con bloques de todos los tipos en modo **simulación** de forma incorrecta	El sistema valida el ejercicio y lo da por incorrecto	Superado
15	**Resolución** de un ejercicio de dos ejercicios tipo pregunta/respuesta: comunidades y provincias introduciendo respuestas correctas e incorrectas	El sistema valida el ejercicio y actualiza el progreso del usuario al finalizar el ejercicio	Superado
16	**Visualización** de panel de usuario	El sistema muestra los ejercicios creados por el usuario y los avances realizados en los ejercicios de tipo pregunta/respuesta	Superado
17	**Visualización** el avance de alumnos habiendo ingresado con una cuenta de profesor y con alumnos asociados.	Se muestra el avance de todos sus alumnos	Superado

Tabla 6.1: Pruebas generales

6.2. Pruebas de integración con el software de control del robot

Una vez asegurado el correcto funcionamiento de la aplicación web se ha procedido a realizar pruebas de integración con el sofware de control del robot. Para realizar estas pruebas, lo primero que se realizó fue la preparación de un escenario real en el que estuviese

marcada la cuadrícula con las que estamos trabajando en el sistema para la generación de coordenadas reales 6.1. De esta forma, podemos comprobar de manera exacta si las coordenadas calculadas y enviadas al software de control se asemejan a la trayectoria teórica.

Una vez el escenario estuvo realizado, se procedió a realizar la comunicación de ambos sistemas.

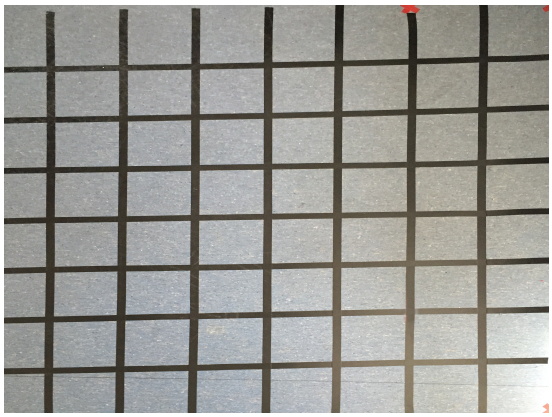


Figura 6.1: Escenario de preparación para la ejecución de ejercicios con un robot

Nº	Descripción	Resultado esperado	Test
1	Ejecución del script que se comunicará con el software de control del robot desde terminal para comprobar el correcto funcionamiento de la comunicación, indicándole la coordenada inicial y los movimientos deseados.	El script traduce los movimientos a coordenadas y se los envía al sistema de control del robot. Este los recibe y ejecuta correctamente las órdenes, trasladando al robot a las coordenadas indicadas.	Superado
2	Ejecución en modo robot sin lenguaje de programación añadido en la vista de control directo del robot	La interfaz se bloquea hasta que tiene respuesta del sistema de control del robot. Una vez ha obtenido respuesta se desbloquea e informa al usuario. El robot se mueve a la casilla 0,0.	Superado
3	Ejecución en modo robot con lenguaje de programación posible añadido en la vista de control directo del robot	La interfaz se bloquea hasta que tiene respuesta del sistema de control del robot. Una vez ha obtenido respuesta se desbloquea e informa al usuario. El robot realiza los movimientos indicados.	Superado
4	Ejecución en modo paso a paso, los movimientos se envían de uno en uno	El robot se traslada de casilla en casilla.	Superado

Nº	Descripción	Resultado esperado	Test
5	Ejecución en modo robot con lenguaje de programación erróneo añadido en la vista de control directo del robot	La interfaz se bloquea hasta que tiene respuesta del sistema de control del robot. Una vez ha obtenido respuesta se desbloquea e informa al usuario. El robot realiza los movimientos indicados hasta el movimiento no posible.	Superado
6	Ejecución en modo robot desde la vista de resolución de un recorrido propuesto, incluyendo el correcto itinerario a seguir.	El robot realiza la ruta correctamente y la interfaz se bloquea hasta que recibe respuesta del sistema de control del robot.	Superado

Tabla 6.2: Pruebas de integración

6.3. Pruebas con usuarios

Las pruebas con usuarios se han realizado con dos niños de 6 y 8 años en las instalaciones de la Universidad Autónoma de Madrid. Las pruebas consistieron en la resolución de distintos ejercicios que se les fueron proponiendo.

Nada más empezar se les preparó un escenario en el que se marcaba el recorrido que debían resolver utilizando la vista de control directo del robot. A continuación se les explicó el funcionamiento de la aplicación de programación, mostrándoles el procedimiento de "drag & drop" de los bloques en el entorno de programación.

Una vez se pusieron manos a la obra se pudo ver que el procedimiento les resultaba muy intuitivo aunque se notó la dificultad que les supone a niños de edades más tempranas el uso del ratón para la selección de los bloques de programación que podría solucionarse con el uso de dispositivos con pantalla táctil. Una vez que resolvieron el ejercicio propuesto, se animaron y ellos mismos se pusieron a diseñar nuevos ejercicios para ir resolviéndolos.

Cabe destacar la curiosidad que les despertó la aplicación que empezó a notarse cuando empezaron a realizar códigos con bucles anidados para descubrir qué pasaba con movimientos no posibles que se salían del tablero o largas series de movimientos.

Una vez terminamos con el control directo del robot, se les propuso los ejercicios propuestos diseñados en la aplicación. En esta vista utilizaron bloques de control más complejos. Para ayudar a entender su funcionamiento, probaron la simuación de código en la aplicación que les indica exactamente en qué movimiento exacto se habían equivocado.

Finalmente, se les propuso la realización de ejercicios personalizados en la aplicación web. El diseño del laboratorio de creación de ejercicios resultó intuitivo, aunque se percibió la dificultad que podía suponer el lenguaje utilizado en los lenguajes de aviso de las reglas de creación de tableros.

6.3.1. Conclusiones generales

El sistema web no mostró ningún problema durante las pruebas con usuarios y demostró una gran robustez en el diseño durante todas las pruebas.

Los únicos cambios que se decidieron realizar fueron en la interfaz gráfica, principalmente los textos explicativos.

En general, los niños no tuvieron problema en el entendimiento de la aplicación, el uso del entorno de programación del robot les resultó muy intuitivo y pudieron entender sin explicación previa el funcionamiento de cada uno.

Un problema con el que nos encontramos durante las pruebas del sistema fue poder mantener a los niños fuera del espacio dedicado al movimiento del robot, por lo que propusimos que los espacios a utilizar estuviesen delimitados.

6.4. Resultados

Tras el desarrollo de este proyecto se ha obtenido como resultado la herramienta "ARDUAMBOT", cuyo principal objetivo es la programación de un robot mediante el uso de un lenguaje de programación por bloques. A continuación, en la figura 6.2 se muestra un ejemplo de la resolución de un ejercicio propuesto por la aplicación. La resolución se realiza seleccionando los bloques de la barra de herramientras y arrastrándolos al panel de programación. Una vez el programa se ha modificado se selecciona la opción de ejecución en el robot.

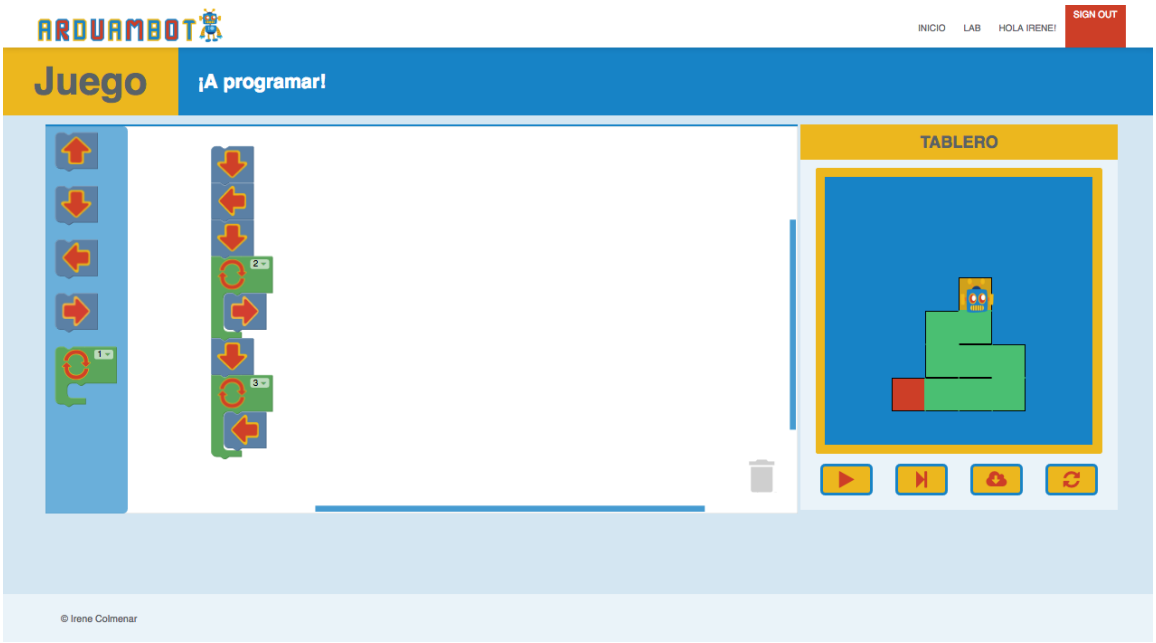


Figura 6.2: Entorno de programación y código realizado

En la figura 6.3 se muestra la animación de la simulación del código en la aplicación y el desplazamiento del robot sobre el tablero real.

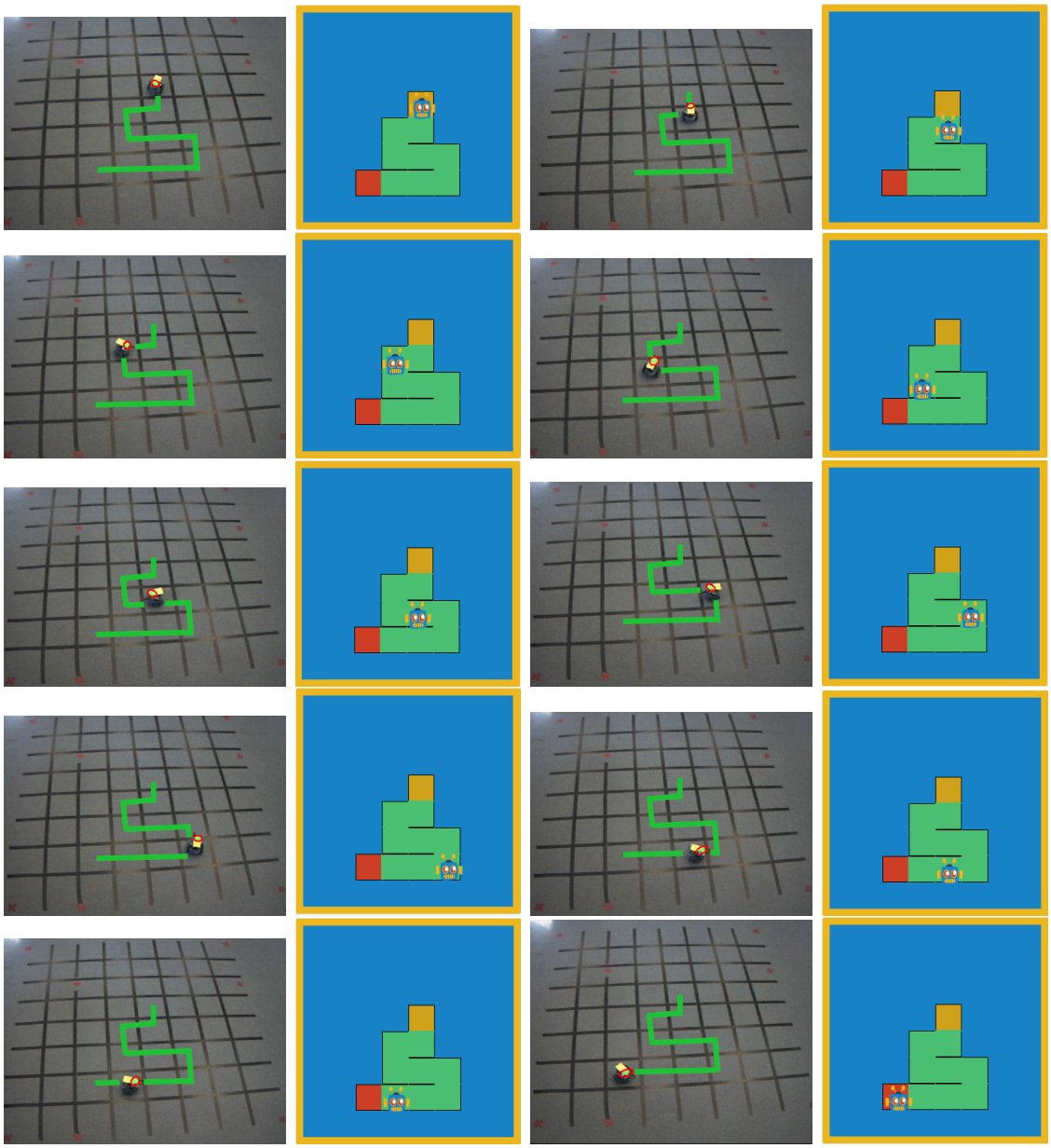


Figura 6.3: Ejecución de código en el robot y simulación en la aplicación

7 | Conclusiones y líneas futuras

El motivo principal que llevó al desarrollo de este proyecto fue la falta de herramientas para introducir a los más pequeños en el mundo de la programación.

Partiendo de este problema, se planteó la realización de un proyecto cuyo objetivo principal fuese el desarrollo de una aplicación para la programación por bloques para un robot Arduino.

Este objetivo se ha considerado logrado. De manera adicional, la aplicación desarrollada se ha convertido en una plataforma que brinda al usuario la capacidad de crear ejercicios propios, de tener un seguimiento de sus propios avances y además de tener un tutor que pueda seguir el avance de todos los alumnos que tenga asignados.

Además, hemos podido ver la parte atractiva que supone para los más pequeños el control de un robot, por lo que el sistema ha aprovechado esa característica para integrarlo en la resolución de ejercicios de un ámbito completamente distinto como es la geografía. Esta integración y planteamiento de nuevos tipos de actividades abren un nuevo frente en cuanto a las oportunidades a las que nos puede llevar.

Tras la realización de pruebas con usuarios de cortas edades hemos podido demostrar la importancia de la usabilidad y el diseño en aplicaciones orientadas a la educación. Para que una herramienta educativa atraiga a un niño es necesario que la página esté adaptada a él, para que así capte su máxima atención.

En cuanto a ideas que podrían realizarse en **líneas futuras**, hemos considerado las siguientes:

- Mostrar en la web una **traducción a lenguaje Arduino** de las instrucciones indicadas en Blockly. Debido a que este proyecto estaba destinado a los más pequeños, no se ha llegado a realizar una traducción del lenguaje de bloques a lenguaje Arduino de una forma visual, pero si se llegase al objetivo de enfocar esta aplicación a usuarios más experimentados sería una buena forma de reorientar la aplicación.
- **Adaptar las actividades adicionales que aprovechan el uso de control de robot** para que el diseño de los ejercicios aproveche las instrucciones de programación en su resolución.
- Mayor integración con el sistema de control de la cámara para la inclusión de nuevos bloques de programación que puedan **interaccionar directamente con el entorno real**.
- Finalmente, podría mejorarse la parte educativa de la aplicación añadiendo la posibilidad **múltiples profesores** por alumno.

Bibliografía

[1] José Alberto González Pérez. *Tendencias en educación en la sociedad de las tecnologías de la información*. 2013. URL: http://www.tribunavalladolid.com/noticias/la-tecnologia-actual-en-nuestra-sociedad/1369849795.

[2] Luis Jesús Padrón Arredondo. *Las Nuevas Tecnologías de la Información (NTIC) en la medicina, la Telemedicina en Cuba*. Universidad Médica de Villa Clara, 2006. URL: http://www.rcim.sld.cu/revista_10/articulos_htm/tecnologiainf.htm.

[3] Erick Fernández Merette. *Implantación de laboratorios TIC y de una asignatura troncal de aprendizaje de codificación Informática en niños de 6 a 8 años en la ciudad de Gaspar Hernández en República Dominicana*. Universidad Politécnica de Madrid, 2013. URL: http://www.dit.upm.es/~posgrado/doc/TFM/TFMs2012-2013/TFM_Erick_Fernandez_2013.pdf.

[4] *Code*. Code.org, 2013. URL: https://code.org/.

[5] *Blockly*. Google, 2016. URL: https://developers.google.com/blockly/.

[6] *Scratch*. MIT Media Lab, 2014. URL: https://code.org/.

[7] *Scratch For Arduino*. Citilab, 2015. URL: http://s4a.cat/index_es.html.

[8] *bibloq*. Mundo Reader, S.L., 2014. URL: http://bitbloq.bq.com/.

[9] *BQ*. Mundo Reader S.L, 2009. URL: http://www.bq.com/.

[10] *LEGO Mindstorms*. The LEGO Group, 2016. URL: http://www.lego.com/es-es/mindstorms.

[11] Iván Márquez Pardo. *Control visual de un robot móvil mediante una cámara cenital*. Universidad Autónoma de Madrid, 2016.

[12] *HTML*. World Wide Web Consortium, 2009. URL: https://es.wikipedia.org/wiki/HTML.

[13] *CSS*. World Wide Web Consortium, 1996. URL: https://en.wikipedia.org/wiki/Cascading_Style_Sheets.

[14] *JavaScript*. Netscape Communications Corp, Mozilla Fundation, 1995. URL: https://developer.mozilla.org/es/docs/Web/JavaScript.

[15] *jQuery*. jQuery Team, 2006. URL: http://jquery.com.

[16] *MySQL*. Oracle Corporation, 1995. URL: http://www.mysql.com.

[17] *PHP*. Zend Technologies, 1995. URL: https://secure.php.net.

[18] *Python*. Python Software Foundation, 1991. URL: https://www.python.orgphp.

[19] *Sublime Text*. Jon Skinner, 2013. URL: http://www.sublimetext.com.

[20] *Adobe Illustrator*. Adobe Systems, 1987. URL: http://www.adobe.com/la/products/illustrator.html.

[21] *Bitbucket*. Atlassian, 2008. URL: https://bitbucket.org/.

[22] *freepik*. Graphic Resources LLCpik, 2010-2015. URL: http://www.freepik.es.

[23] *Font Awesome*. Dave Gandy, 2012. URL: http://fontawesome.io.

[24] *Raphaël—JavaScript Library*. Sencha Labs, 2008. URL: http://dmitrybaranovskiy.github.io/raphael/.

[25] *SVG*. w3c, 2001. URL: http://www.w3.org/Graphics/SVG/.

[26] *c3.js*. sencha, 2011. URL: http://c3js.org.

[27] *D3.js*. BSD, 2011. URL: http://c3js.org.

[28] Yusef Hassan Montero. *Diseño web orientado a niños*. 2004. URL: http://www.nosolousabilidad.com/articulos/diseno_orientado_ninos.htm?ifra.

Apéndices

A | Estructura del proyecto

El árbol que describe la estructura del proyecto sólo especifica la jerarquía de los directorios desarrollados para este proyecto, excluyendo las bibliotecas externas utilizadas y especificadas anteriormente.

```
/arduambot
├── blockly
├── blockui-master
├── c3
├── css
│   ├── animation.css
│   ├── general.css
│   ├── index.css
│   ├── lab.css
│   ├── level.css
│   ├── levels.css
│   ├── login-register.css
│   ├── trivial.css
│   └── user.css
├── font-awesome
├── img
├── js
│   ├── animation.js
│   ├── dom.js
│   ├── general.js
│   ├── level.js
│   ├── login-register.js
│   ├── trivial.js
│   └── user.js
└── php
    ├── addAlumn.php
    ├── delAlumn.php
    ├── deleteBoard.php
    ├── insertDB.php
    ├── insertMapProgress.php
    ├── insertProgress.php
    ├── log-out.php
    ├── login.php
    ├── register.php
    └── remote_control.php
```

```
/
    ├── userProgress.php
├── python
    ├── receptor.py
    ├── trayectoria.py
├── spain-map
├── view
    ├── header.php
    ├── lab.php
    ├── level.php
    ├── levels.php
    ├── login-register.php
    ├── trivial.php
    ├── user.php
└── index.php
```

B | Maquetas de las vistas

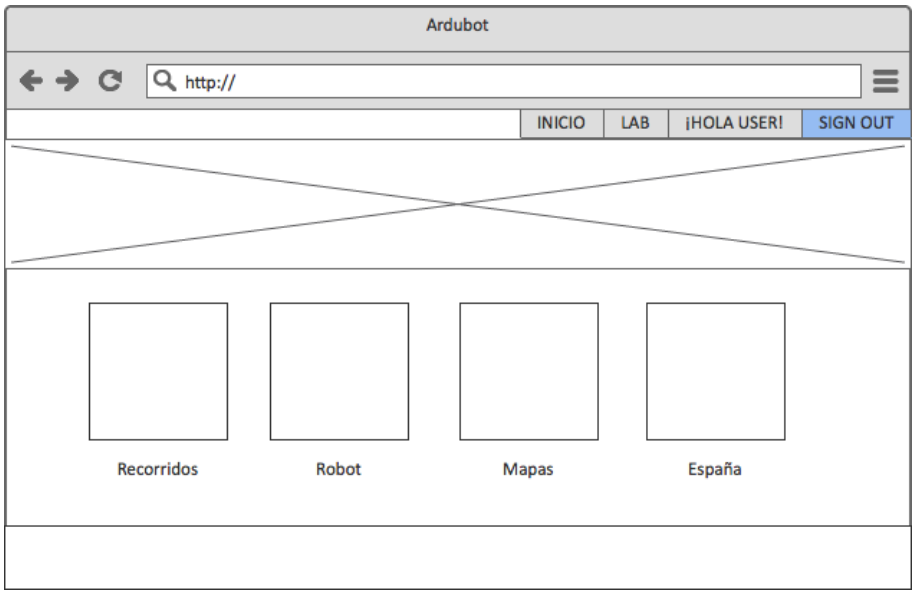


Figura B.1: Maqueta de la página principal

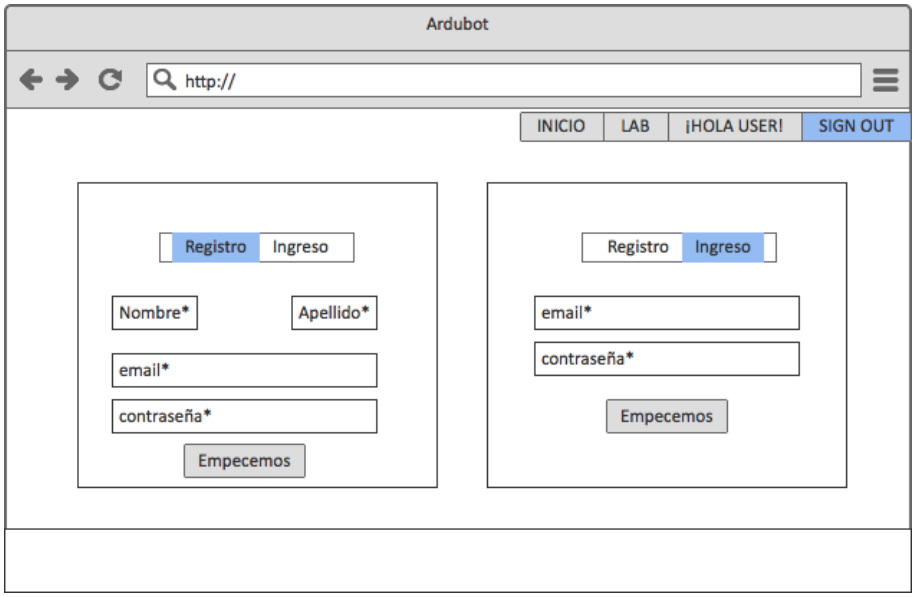


Figura B.2: Maqueta de la vista de registro e ingreso

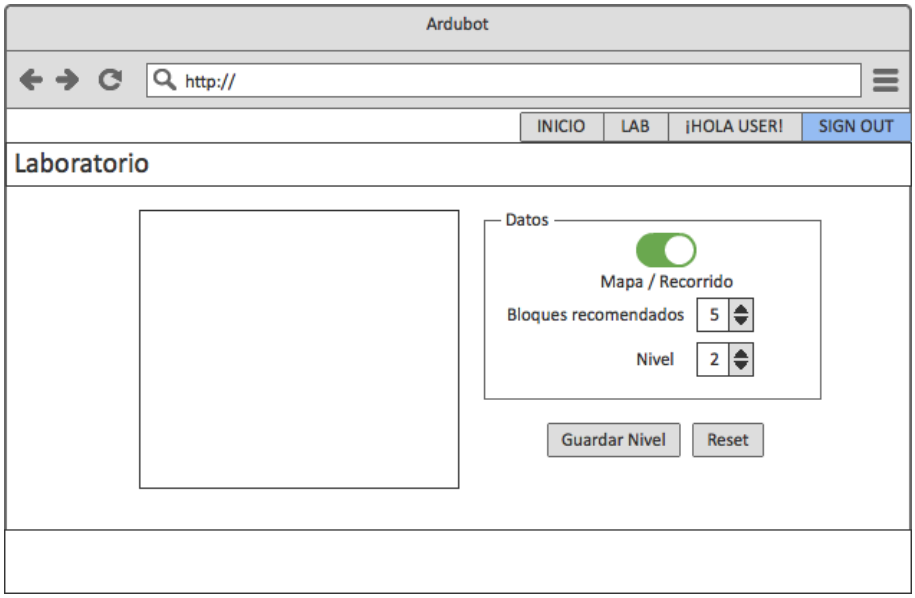


Figura B.3: Maqueta de la vista del laboratorio

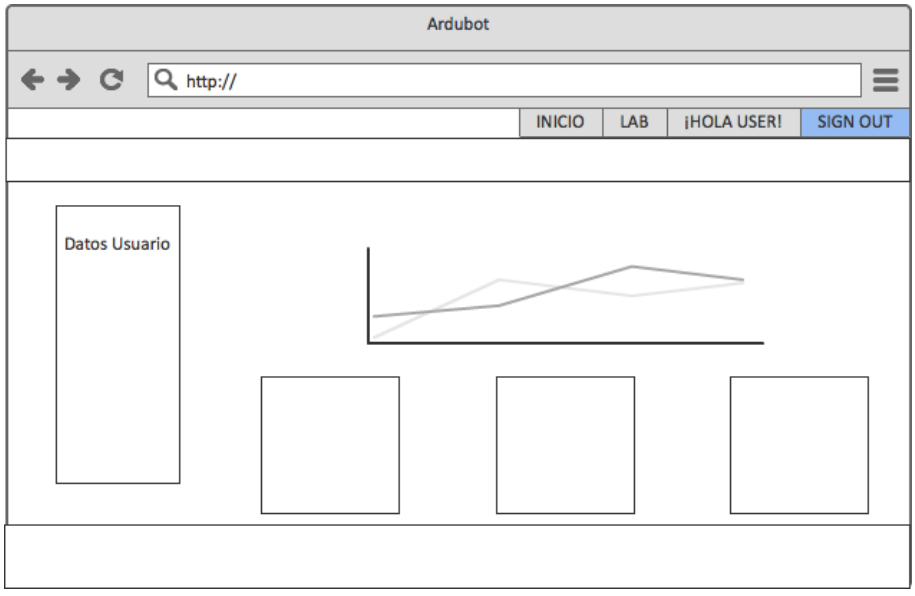


Figura B.4: Maqueta de la vista de usuario

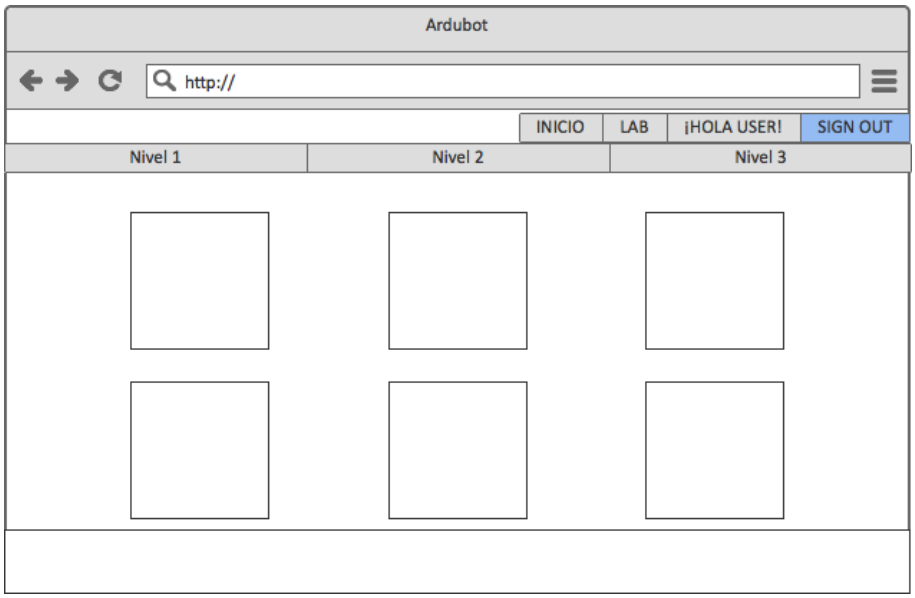


Figura B.5: Maqueta de la vista del panel de niveles

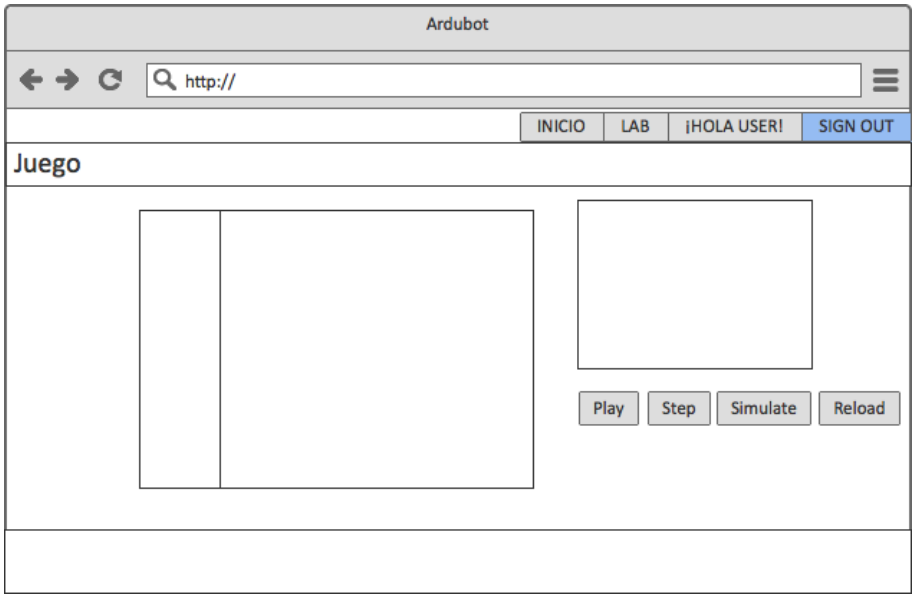


Figura B.6: Maqueta de la vista de recorridos y mapas

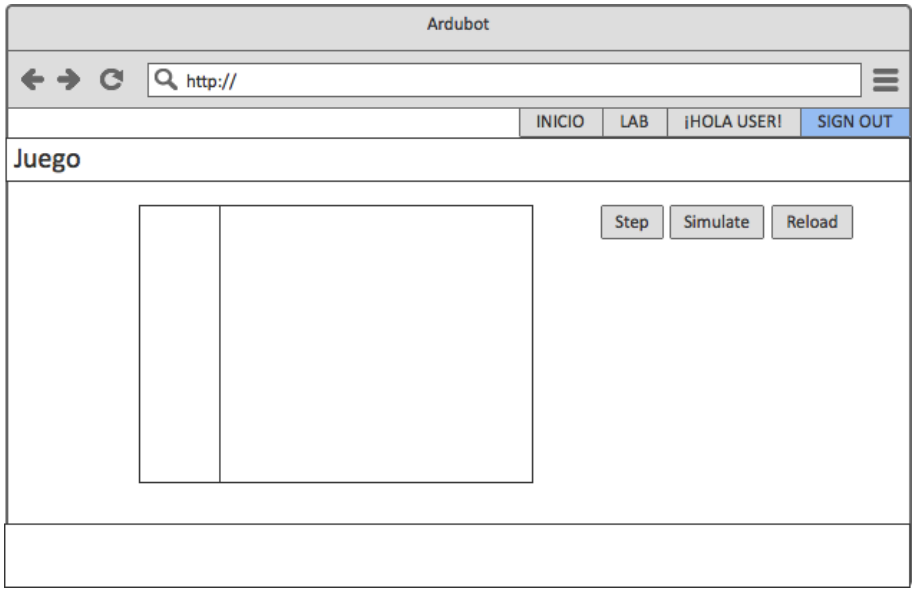


Figura B.7: Maqueta de la vista de control directo del robot

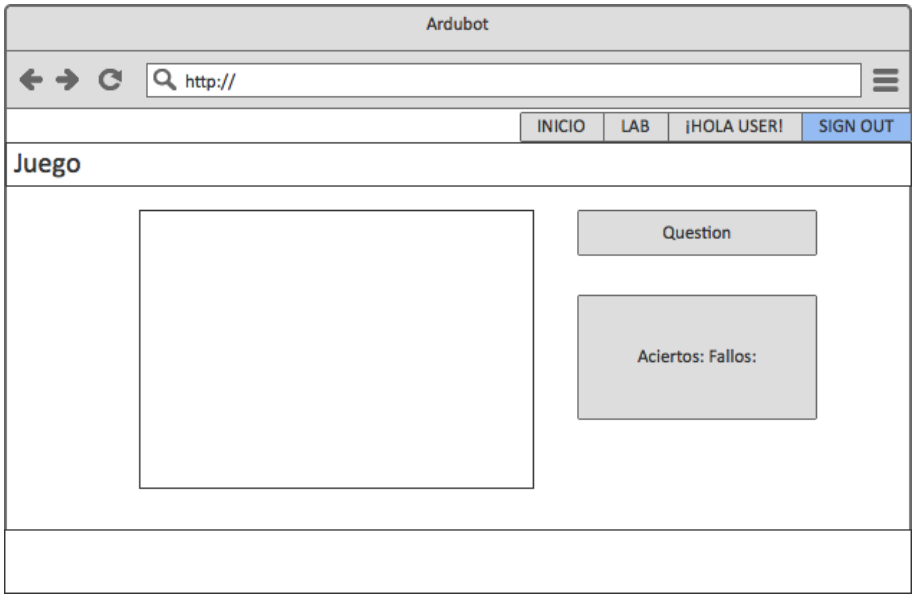


Figura B.8: Maqueta de la vista mapas geográficos

C | Vistas finales

Figura C.1: Vista de la página principal

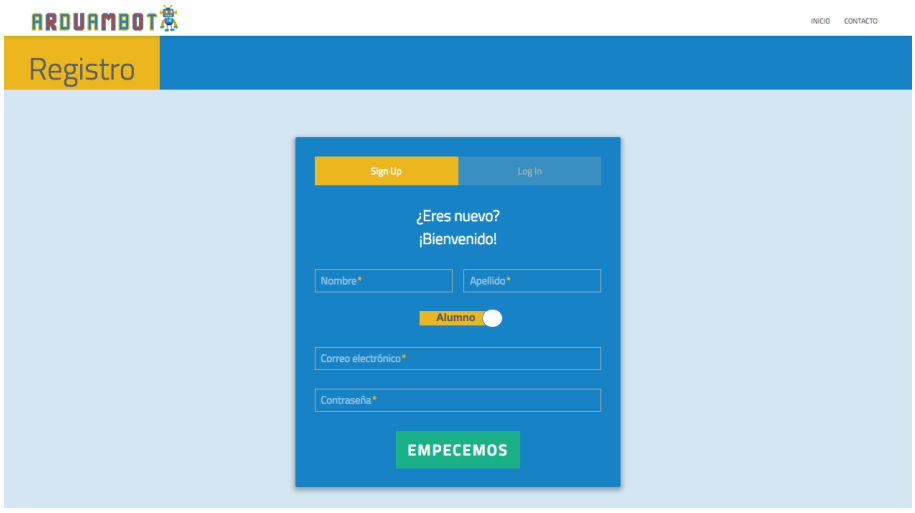


Figura C.2: Vista del registro e ingreso de usuario

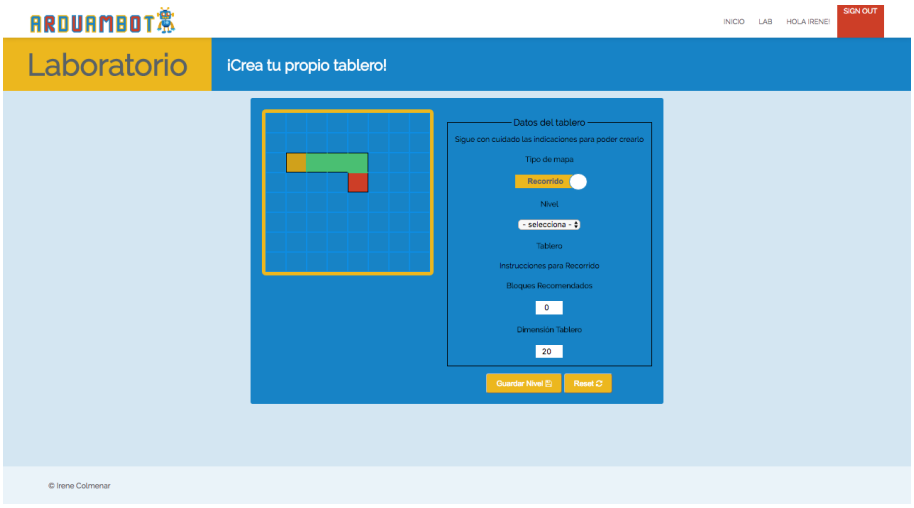


Figura C.3: Vista del laboratorio de creación de ejercicios

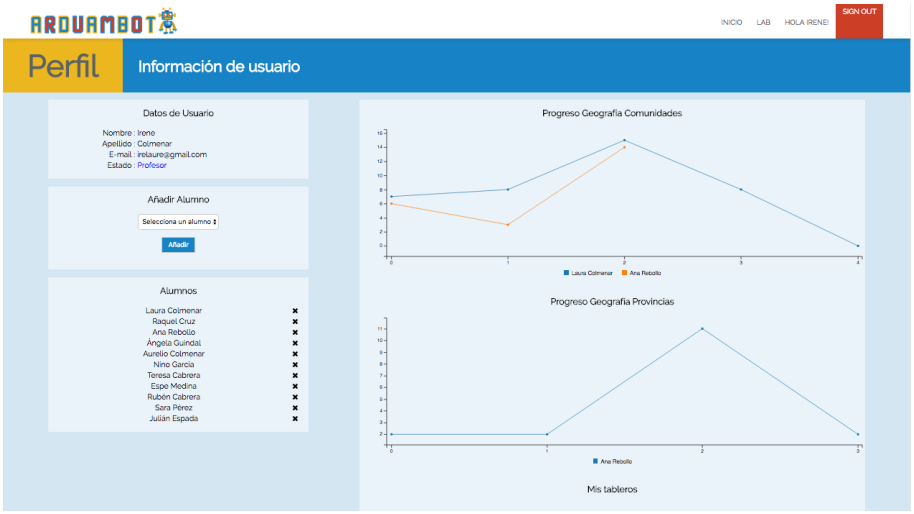


Figura C.4: Vista superior del panel de usuario

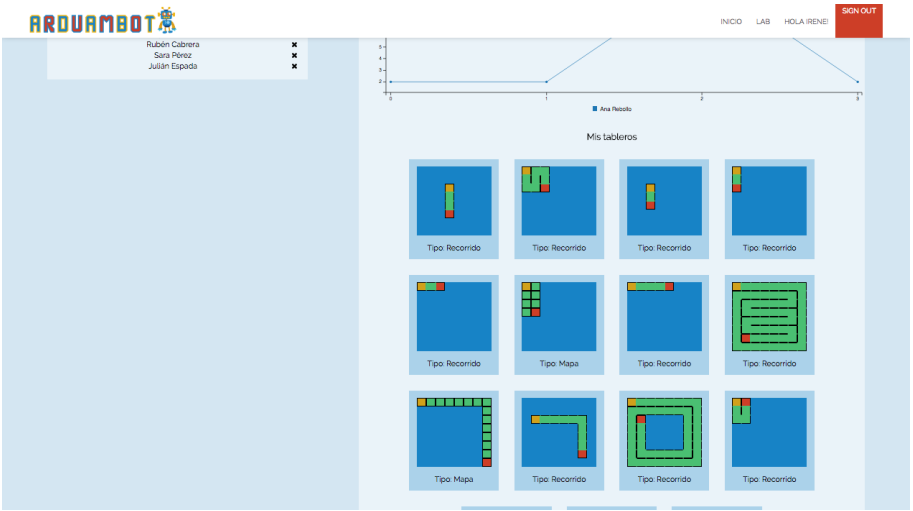


Figura C.5: Vista inferior del panel de usuario

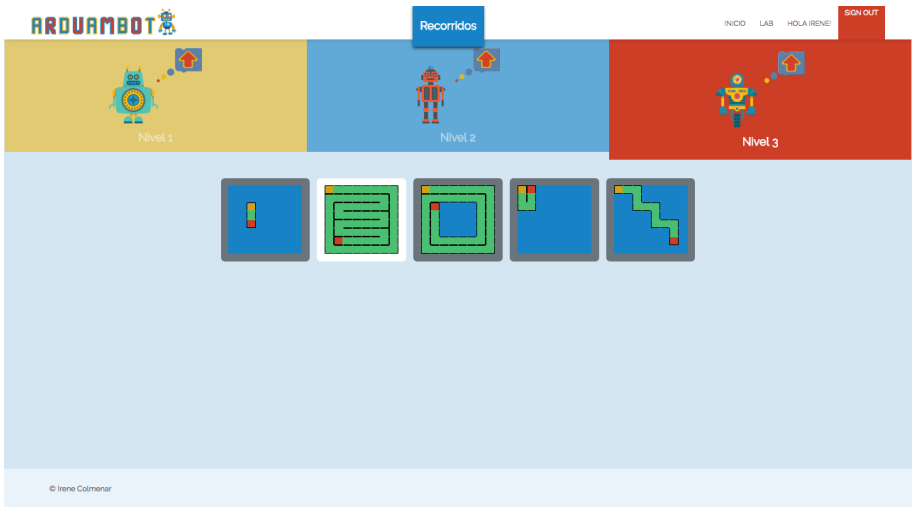


Figura C.6: Vista del panel de niveles

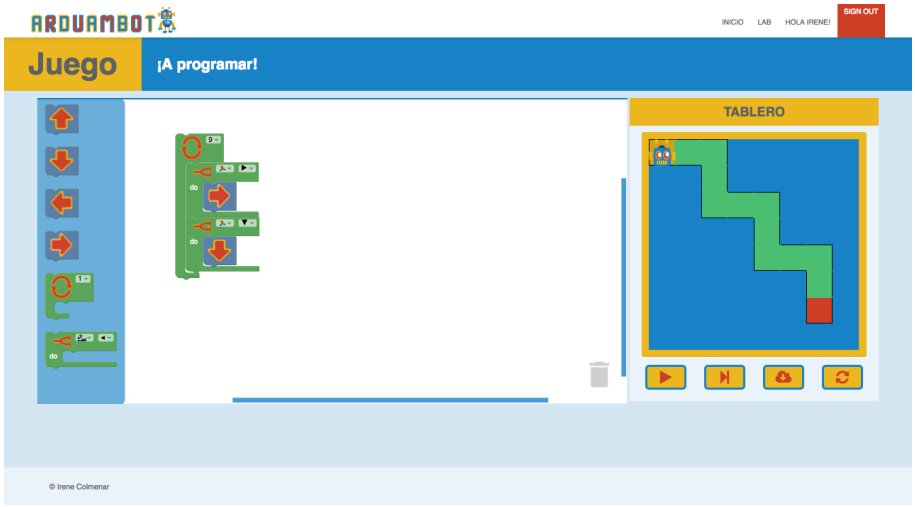


Figura C.7: Vista de recorridos y mapas

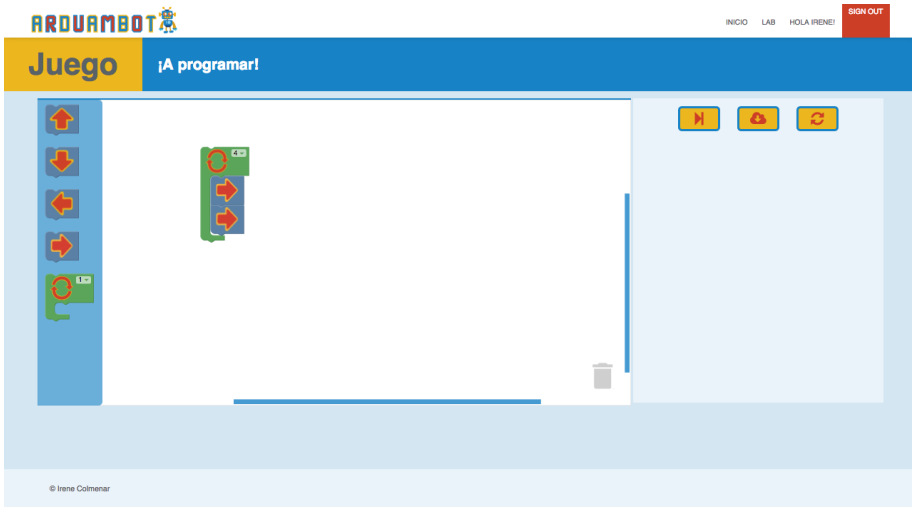


Figura C.8: Vista de control directo del robot

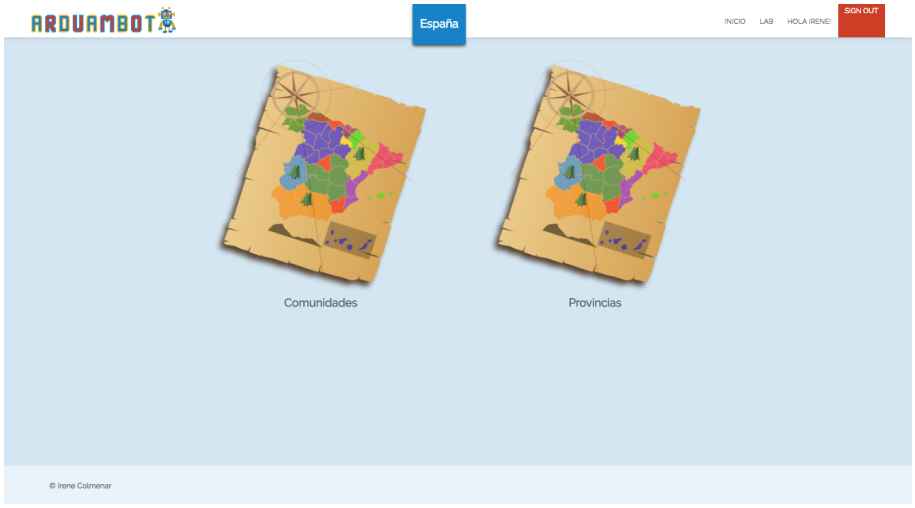


Figura C.9: Vista menú mapas geográficos

Figura C.10: Vista mapas geográficos